Fast Convergence of Regularised Region-based Mixture of Gaussians for Dynamic Background Modelling

Varadarajan, S., Wang, H., Miller, P., & Zhou, H. (2015). Fast Convergence of Regularised Region-based Mixture of Gaussians for Dynamic Background Modelling. *Computer Vision and Image Understanding*, *136*, 45-58. https://doi.org/doi:10.1016/j.cviu.2014.12.004

Published in:
Computer Vision and Image Understanding

Document Version:
Peer reviewed version

Queen's University Belfast - Research Portal:
Link to publication record in Queen's University Belfast Research Portal

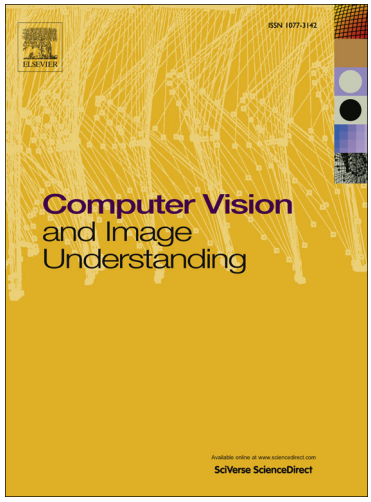
Publisher rights
© 2015, Elsevier. Licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International http://creativecommons.org/licenses/by-nc-nd/4.0/ which permits distribution and reproduction for non-commercial purposes, provided the author and source are cited.

General rights
Copyright for the publications made accessible via the Queen's University Belfast Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The Research Portal is Queen's institutional repository that provides access to Queen's research output. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact openaccess@qub.ac.uk.

Download date:05. May. 2023

Accepted Manuscript



Fast Convergence of Regularised Region-based Mixture of Gaussians for Dynamic Background Modelling

Sriram Varadarajan, Hongbin Wang, Paul Miller, Huiyu Zhou

PII: S1077-3142(14)00242-2
DOI: http://dx.doi.org/10.1016/j.cviu.2014.12.004
Reference: YCVIU 2199

To appear in: *Computer Vision and Image Understanding*

Received Date: 15 April 2014
Accepted Date: 17 December 2014

Please cite this article as: S. Varadarajan, H. Wang, P. Miller, H. Zhou, Fast Convergence of Regularised Region-based Mixture of Gaussians for Dynamic Background Modelling, *Computer Vision and Image Understanding* (2014), doi: http://dx.doi.org/10.1016/j.cviu.2014.12.004

This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

ACCEPTED MANUSCRIPT

Fast Convergence of Regularised Region-based Mixture of Gaussians for Dynamic Background Modelling

Sriram Varadarajan[a,*], Hongbin Wang[a], Paul Miller[a], Huiyu Zhou[a]

[a] *The Centre for Secure Information Technologies (CSIT), Queen's University Belfast, Belfast BT3 9DT, United Kingdom*

Abstract

The momentum term has long been used in machine learning algorithms, especially back-propagation, to improve their speed of convergence. In this paper, we derive an expression to prove the $O(1/k^2)$ convergence rate of the online gradient method, with momentum type updates, when the individual gradients are constrained by a growth condition. We then apply these type of updates to video background modelling by using it in the update equations of the Region-based Mixture of Gaussians algorithm. Extensive evaluations are performed on both simulated data, as well as challenging real world scenarios with dynamic backgrounds, to show that these regularised updates help the mixtures converge faster than the conventional approach and consequently improve the algorithms performance.

Keywords: Momentum, Convergence Rate, Generative Models, Region-based Mixture of Gaussians, Dynamic Background Modelling

*Corresponding author

Email addresses: svaradarajan01@qub.ac.uk (Sriram Varadarajan), h.wang@ecit.qub.ac.uk (Hongbin Wang), p.miller@qub.ac.uk (Paul Miller), h.zhou@ecit.qub.ac.uk (Huiyu Zhou)

ACCEPTED MANUSCRIPT

1. Introduction

Generative models have been extensively applied to many areas of computer vision. One of the most intensive areas of application, that has seen a lot of activity in this regard, is that of background modelling for foreground detection in surveillance video [1]. In particular, the Mixture of Gaussians (MoG) model proposed by Stauffer and Grimson [2] has, it could be argued, become the standard de facto approach to generative background modelling. From a Bayesian perspective, the online MoG algorithm uses a Robbins-Monro type stochastic approximation technique [3] to update the mixture parameters over time. These updates can be controlled by a learning rate parameter that can be tuned to vary the rate of learning of the mixture model. From a machine learning perspective, these updates can also be viewed as analogous to widely-used gradient descent methods.

Whilst the MoG algorithm has been shown to work well for static background scenes, it has struggled to cope with scenes whose background, or part thereof, is dynamic. Sources of dynamism include moving backgrounds, such as the surface of water and vegetation blowing in the wind, and also camera movements due to vibration, or the fact it is on-board a moving vehicle, such as a bus or train. From a modelling perspective there are two main differences between static and dynamic backgrounds. Firstly, in a static background scenario, it is assumed that the source of a sample distribution has a one-to-one mapping with a given pixel. However, for the dynamic scenario, the mapping is one-to-many, i.e., the samples for a specific distribution may be captured over a region of the imaging sensor. Varadarajan et al. [4] previously addressed this issue by developing a region-based MoG

ACCEPTED MANUSCRIPT

(RMoG) algorithm which showed significant performance improvements over the standard MoG algorithm for dynamic backgrounds. Furthermore, we formally derived the update equations using a theoretical framework for which the standard MoG is a special case. Secondly, the number of samples acquired for a distribution may be much smaller in the dynamic case.

A drawback of gradient descent methods in relation to the second issue, is that they converge slowly [5]. One way to speed up convergence is by introducing an additional momentum term to the update equation, which adds a fraction of the difference between the two previous values of the parameters. The additional term can be viewed as inertia as it takes a small step further in the same direction of the previous update, hence the name heavy-ball. This reduces oscillations when the direction of the gradient keeps changing. It can also aid the learning rate by pushing the update further towards the optimum value if there is no change in the direction of the gradient, which helps to increase the speed of convergence to the optimum value. Momentum can also help the system escape local minima better than conventional gradient methods.

Wang and Miller [6] introduced the use of the momentum term for regularising the classification Expectation-Maximisation (EM) based MoG algorithm and presented some initial results for background modelling in video scenes that showed promise. In this work, we incorporate a momentum term into our region-based framework, in order to address the second issue, outlined above, in relation to dynamic backgrounds. Finally, important properties for learning algorithms are whether they are guaranteed to converge and, if so, their rate of convergence. Previously, a rate of convergence of $O(1/k^2)$

ACCEPTED MANUSCRIPT

has been proven for the batch version of heavy-ball [7]. Also, Varadarajan et al. provided a proof of convergence for the online version of this algorithm in [8]. While this shows that the method converges, the rate of convergence was not given. Therefore, the second major contribution of this work is to prove a rate of convergence of $O(1/k^2)$ for the online algorithm.

The remainder of this paper is structured as follows. We highlight some of the related work in literature in section 2. The spatio-temporal modelling approach with RMoG algorithm is briefly introduced in section 3. The $O(1/k^2)$ rate of convergence for the online gradient method with a momentum term is derived in section 4. This extrapolation step is then applied to the RMoG framework to derive a new background subtraction algorithm which is explained in detail in section 5. Section 6 contains a comprehensive evaluation of the algorithm using simulated data and well-known real video sequences with dynamic backgrounds, whilst comparing it with other state-of-the-art MoG variants. The conclusion is presented in section 7.

2. Related Work

Statistical modelling of pixels for the purpose of motion detection has been widely researched over recent years. For a comprehensive survey on this topic, refer to [9]. We mention some of the landmark papers in this field here. Wren et al.[10] used a single Gaussian distribution for background modelling in their real-time human body tracking system called Pfinder. Friedman and Russell [11] were the first to model pixels in a traffic surveillance scene as a mixture of three Gaussian distributions for shadows, road and vehicles. They built the MoG model by using an incremental version of the EM algorithm

ACCEPTED MANUSCRIPT

[12]. Their prior model is, however, heuristic and assumes that the vehicle pixels are brighter, in general, than road pixels and that the shadow pixels are the darkest of the three classes. These assumptions are somewhat restrictive and are not always satisfied. McKenna et al [13] developed an adaptive version of the EM algorithm using a local batch of previous L frames at each time instant. To reduce the computational cost, they used a recursive method to estimate the values.

Stauffer and Grimson [2] used a mixture of Gaussian distributions to model backgrounds, but as described above, this algorithm suffered when the scene has highly dynamic components such as a water surface or waving trees. There have been numerous extensions proposed to this approach. Kaewtrakulpong and Bowden [14] revisited the update equations for the mixture parameters and proposed using two different sets of equations, one during the initial learning phase and the other during the maintenance phase. During the learning phase, they used an incremental version of the EM algorithm to update the parameters based on the expected sufficient statistics and then switching to L-recent window update equations [13] after the initial L frames. They showed that this initial estimation of the parameters allows for a faster convergence to a stable background model. However, this two phase schedule does not help in learning newer objects in the scene that arrive after the learning period. In order to overcome this problem, Lee [15] blended the two phases into one by gradually moving from the incremental EM phase to a recursive filter phase every time a mixture is created or reassigned. Zivkovic [16] proposed using a Dirichlet prior to dynamically estimate the number of Gaussians. Moving away from parametric distributions, Elgammal et al [17]

ACCEPTED MANUSCRIPT

introduced a non-parametric modelling of pixels using Kernel Density Estimation. These methods are all pixel independent approaches, hence they do not completely address the issue of dynamic backgrounds.

Greenspan et al. [18] proposed a unsupervised clustering method by modelling a group of frames together with a Gaussian mixture model rather than updating them at every time instant. A spatial neighbourhood between pixels was considered for modelling by Sheikh and Shah in [19]. However, they had to maintain a history of frames resulting in a high dimensionality problem. Jodoin et al. [20] assumed that the spatial variations over a region were the same as the temporal variations of an individual pixel and built a combined spatial and temporal framework. However, this assumption is not always true, as explained by the authors themselves with the help of a pertinent example. Zhang et al [21] introduced the non-parametric version of the spatial-temporal model for modelling pixels. As in the case of most non-parametric techniques, it suffers from high memory requirements as it requires maintaining samples from a series of previous frames. Dalley et al. [22] proposed a heuristic generalization of the MoG model introducing four different types of updates based on hard and soft assignments of the clusters. Yu et al. [23] used two spatial-color Gaussian mixture models (SCGMM) in combination with a graph cut algorithm to track large moving objects in a scene. They used the EM algorithm to combine and split the two SCGMM over successive frames in order to track both the foreground and the background regions together under the assumption that the colours do not change in the region. However, this assumption is only suitable for large uniform objects in the scene.

ACCEPTED MANUSCRIPT

The Momentum term in gradient descent method was first introduced by Polyak [24] and was called the heavy-ball approach. The heavy-ball method is a batch version of the gradient descent algorithm with an additional momentum term that accelerates convergence. An optimal version of the heavy-ball method was derived by Nesterov [7]. He obtained a superior convergence rate of $O(1/k^2)$ compared to the gradient descent method which had a convergence rate of $O(1/k)$. This is the fastest convergence rate achievable by a first-order method. Beck and Teboulle [25] extended this algorithm to composite functions by using proximal gradient updates known as FISTA, this algorithm is used widely in image processing for image deblurring and deconvolution. In 2008, Paul Tseng combined all these algorithms and more into a unified framework for accelerated proximal gradient algorithms [26]. All these algorithms are for batch learning, but we show that similar benefits can be gained by using the momentum term in online learning.

Rumelhart et al. [27] studied the inclusion of a momentum term to improve the convergence rate of the learning scheme in the context of back-propagation neural networks. Momentum updating has also been applied previously in the Least Mean Squares (LMS) algorithm by Shynk and Roy [28]. Bertsekas [29] proposed a hybrid class of methods combining the LMS and steepest descent methods for a fast convergence while training large datasets. Zhang [30] discussed the convergence results for the momentum term in a two-layer feedforward network.

Acceleration for the purpose of fast convergence has also been used before in various Computer Vision applications. For instance, Bao et al. [31] used the accelerated proximal gradient method [26] for solving the mini-

ACCEPTED MANUSCRIPT

mization function in their real time tracking system. Conditional Random Fields (CRF) were trained using an accelerated version of Stochastic Gradient Descent by Viswanathan et al. [32] and applied to image denoising and segmentation. In their approach, the step size was adapted by a schedule based on second order statistics. Other works that have used the momentum term to accelerate convergence include object tracking by Le et al. [33], image segmentation by Andersson et al. [34], image deconvolution by Wang and Miller [35], and scene labelling and denoising by Domke [36].

In the context of foreground detection, a simulated annealing technique was used by Li et al. [37] for accelerating the convergence in the post-processing step of their foreground detection algorithm based on Kernel Density Estimation. Guyon et al. [38] also used a proximal gradient method for solving the convex optimization problem of Robust Principal Components Analysis for their foreground detection algorithm. This method is however a offline version that processes a large batch of frames at a time. Lee [15] proposed a learning rate schedule that was computed based on the contributions of each Gaussian component in his background subtraction algorithm. A regularisation term, based on the l_1 norm, was used to introduce sparsity on the inverse Covariance matrices and help reduce dimensionality in the case of high-dimensional Gaussian Mixture Models. In this work, the regularisation term is not required for the dimensionality reduction, rather it is used for the fast convergence of the mixtures. Greggio et al [39] used a modified version of the EM algorithm, using a Dirichlet prior, for the initial learning stage and subsequently used Lee's background subtraction procedure.

In this paper, we derive an expression to show the $O(1/k^2)$ rate of con-

ACCEPTED MANUSCRIPT

vergence for the online gradient method with momentum based updates. We then propose a regularisation term for the RMoG algorithm [4] in the form of momentum updating, to accelerate its convergence rate.

3. Region-based Mixture of Gaussians algorithm (RMoG)

Here, we start by outlining the online RMoG algorithm for dynamic background that was introduced in [4]. This algorithm is a complete framework for MoG modelling extending the standard per-pixel approach to larger regions, thus enabling it to model highly dynamic regions in the scene. The standard MoG algorithm considers pixels to be independent of one another, hence important spatial cues in the scene where there is dynamic motion, such as waves rippling or leaves swaying, are not captured. The RMoG algorithm takes into consideration this spatial relationship between pixels in a region and builds the model accordingly. It was shown in [4] that this framework models dynamic motion in a scene quite effectively.

The RMoG model can be written in mathematical terms as

$$p(\boldsymbol{y}_{i,j}|\Theta) \propto \sum_{q \in \mathcal{R}_{i,j}} \sum_{h=1}^{H} \omega_{qh} * \mathcal{N}(\boldsymbol{y}_{i,j}|\mu_{qh}, \Sigma_{qh}) \tag{1}$$

where $\boldsymbol{y}$ is the pixel (or a feature vector of pixels) at location (i, j) under observation, $\mathcal{N}(\bullet)$ denotes a Gaussian distribution and the parameter set is $\Theta = \{\omega, \mu, \Sigma\}$ namely the weight, mean and variance of each mixture in the distribution. $\mathcal{R}_{i,j}$ denotes the neighbourhood of the feature vector $\boldsymbol{y}$ denoted by $\mathcal{R}_{i,j} = \{i, j : i - r/2 \leq i \leq i + r/2 - 1, j - r/2 \leq j \leq j + r/2 - 1\}$. This usually includes r×r pixels (or feature blocks) around $\boldsymbol{x}$. The subscript q indicates the location of the chosen mixture component in the neighbourhood

ACCEPTED MANUSCRIPT

$\mathcal{R}$. The algorithm is also customisable in terms of the feature and region size to consider for modelling. For instance, a block of pixels can also be considered as a feature instead of the conventional approach of using a single pixel within a given region. It can be easily shown that by using a single pixel as a feature in a 1×1 window, the algorithm reduces to the standard MoG approach.

The mixture parameters are updated by using the following equations:

$$\mu_{k,l,h}^{(t)} = (1-\rho)\,\mu_{k,l,h}^{(t-1)} + \rho\left(y^{(t)}\right) \tag{2}$$

$$\Sigma_{k,l,h}^{(t)} = (1-\rho)\,\Sigma_{k,l,h}^{(t-1)} + \rho\Big(y^{(t)} - \mu_{k,l,h}^{(t-1)}\Big)^2 \tag{3}$$

$$\omega_{k,l,h}^{(t)} = (1-\alpha)\,\omega_{k,l,h}^{(t-1)} + \alpha \tag{4}$$

where (k,l) is the location of the chosen mixture in the neighbourhood and ρ is the rate at which the parameters evolve over time. It is proportional to α and it is varied according to the weight of the mixture.

4. Rate of Convergence for Online Gradient Method with Momentum

In this section an expression is derived to prove the $O(1/k^2)$ convergence rate for the online gradient method with a momentum term. This algorithm can be written as

$$x_{k+1} = y_k - \alpha \nabla f_{\omega_k}(y_k) \tag{5}$$

$$y_k = x_k + \beta_k(x_k - x_{k-1}) \tag{6}$$

Here, x_k is the estimate at instant k, α is the learning rate, β is the momentum parameter and ω_k is the sample randomly drawn in a uniform

ACCEPTED MANUSCRIPT

manner from a set of N data samples. The proof of convergence of this algorithm can be derived by using the Supermartingale Convergence Theorem [8]. For completeness, we include the derivation of this proof in the appendix.

Standard convergence analysis used in batch algorithms is followed here, more specifically the heavy-ball extension for batch gradient descent [40], and error terms are defined for the individual gradient terms compared to the full gradient of the batch algorithm. Then, a growth condition is imposed on these individual gradients similar to [41] who applied this approach to the stochastic gradient descent algorithm. This growth condition was also used in [42] for a deterministic version of the incremental gradient method with a momentum term. In our method, we show the convergence in expectation for the online version of the algorithm.

The derivation is structured in the following manner. The algorithm is first rewritten using the error terms. Then, the function at the $(k+1)^{th}$ iterate given by $f(x_{k+1})$ is bound by assuming that the function f has a Lipschitz continuous gradient. Following this, a term z_k is introduced as a linear combination of two successive iterates, and substituted in the above mentioned bounding equation. Then, taking the expectations of the function and the error terms, a bound is provided for the difference in the expectation of the function at the $(k+1)^{th}$ iterate and the function at x^* by a term depending on the parameter k, thus giving an expression for the convergence rate in $O(1/k^2)$.

Following assumptions used in the derivation of the rate of convergence of the batch algorithm [40], we have

$$\beta_k = \frac{\theta_k(1-\theta_{k-1})}{\theta_{k-1}}, \qquad k = 0, 1, ...$$

ACCEPTED MANUSCRIPT

where the sequence $\{\theta_k\}$ satisfies $\theta_0 = \theta_1 \in (0, 1]$ and

$$\frac{1-\theta_{k+1}}{\theta_{k+1}^2} \leq \frac{1}{\theta_k^2}, \qquad \theta_k \leq \frac{2}{k+2}, \qquad k = 0, 1, ...$$

One possible choice is

$$\beta_k = \frac{k-1}{k+2}, \qquad \theta_k = \frac{2}{k+2}$$

The stochastic nature of the algorithm results in an error e_k in the calculation of the gradient $\nabla f_{\omega_k}(y_k)$ at each step

$$e(k) = \nabla f_{\omega_k}(y_k) - \nabla f(y_k), \tag{7}$$

where $\nabla f(y_k)$ is the full gradient of the function.

It is assumed that the individual gradients are bounded by a linear function of the average gradient, i.e.

$$\max_{\omega_k} \{\|\nabla f_{\omega_k}(y)\|\} \leq C \|\nabla f(y)\| \tag{8}$$

This is a strong growth condition for the individual gradients and has been used previously [41, 42] for both stochastic and deterministic methods.

The online gradient with momentum algorithm can be rewritten in terms of the error from (7) as

$$x_{k+1} = y_k - \alpha(\nabla f(y_k) + e_k) \tag{9}$$

Since the gradient calculation is performed with each sample being uniformly drawn from a set, it means that the expectation of the individual gradient at each iteration is equal to the full gradient at that iteration.

$$E[\nabla f_{\omega_k}(y_k)] = \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(y_k) = \nabla f(y_k) \tag{10}$$

ACCEPTED MANUSCRIPT

Now, similar to [41], the expectation of the error term and the squared error term can then be obtained from (7), (8) and (10) as

$$E[e_k] = 0 \tag{11}$$

$$E[\|e_k\|^2] \leq (C^2 - 1)\|\nabla f(y_k)\|^2 \tag{12}$$

Convergence analysis is then performed based on the assumptions that the function $f(x)$ is smooth and convex for all x giving us a lower bound on f, i.e.

$$f(a) \geq f(b) + \nabla f(b)(a - b) \qquad \forall a, b \tag{13}$$

where the function $f(x)$ has a Lipschitz continuous gradient upper bounded by the constant L, i.e.

$$f(a) \leq f(b) + \nabla f(b)(a - b) + \frac{L}{2}\|a - b\|^2 \tag{14}$$

Applying the quadratic upper bound condition in this case gives us

$$f(x_{k+1}) \leq f(y_k) + \nabla f(y_k)(x_{k+1} - y_k) + \frac{L}{2}\|x_{k+1} - y_k\|^2 \tag{15}$$

Using the definition of x_{k+1} from (9) in (15)

$$f(x_{k+1}) \leq f(y_k) + \nabla f(y_k)(-\alpha(\nabla f(y_k) + e_k)) + \frac{L}{2}\|-\alpha(\nabla f(y_k) + e_k)\|^2 \tag{16}$$

$$= f(y_k) - \alpha\left(1 - \frac{\alpha L}{2}\right)\|\nabla f(y_k)\|^2 - \alpha(1 - \alpha L)\nabla f(y_k)e_k + \frac{\alpha^2 L}{2}\|e_k\|^2 \tag{17}$$

ACCEPTED MANUSCRIPT

Now, using the lower bound property of convex functions (14) and rearranging the terms,

$$f(y_k) \leq (1-\theta_k)f(x_k) + \theta_k f(x^*) + \nabla f(y_k)^T(y_k - (1-\theta_k)x_k - \theta_k x^*) \quad (18)$$

Substituting for $f(y_k)$ from (18) in (17),

$$f(x_{k+1}) \leq (1-\theta_k)f(x_k) + \theta_k f(x^*) + \nabla f(y_k)^T(y_k - (1-\theta_k)x_k - \theta_k x^*) - \alpha\left(1 - \frac{\alpha L}{2}\right)\|\nabla f(y_k)\|^2 - \alpha(1-\alpha L)\nabla f(y_k)e_k + \frac{\alpha^2 L}{2}\|e_k\|^2 \quad (19)$$

By using the relationship between x_k and y_k from (6),

$$y_k = \theta_k z_k + (1-\theta_k)x_k \quad (20)$$

where z_k is a linear combination of successive iterates x_k and x_{k-1}.

Now, by rearranging the terms in equation (20), the relationship between z_k and z_{k+1} can be obtained as

$$z_{k+1} = z_k - \frac{\alpha}{\theta_k}(\nabla f(y_k) + e_k) \quad (21)$$

Using (20) to replace $y_k - (1-\theta_k)x_k$ in equation (19)

$$f(x_{k+1}) \leq (1-\theta_k)f(x_k) + \theta_k f(x^*) + \theta_k \nabla f(y_k)^T(z_k - x^*) - \alpha\left(1 - \frac{\alpha L}{2}\right)\|\nabla f(y_k)\|^2 - \alpha(1-\alpha L)\nabla f(y_k)e_k + \frac{\alpha^2 L}{2}\|e_k\|^2 \quad (22)$$

Now, we can replace the $\theta_k \nabla f(y_k)^T(z_k - x^*)$ term from the above equation

ACCEPTED MANUSCRIPT

by introducing the following expression using (21).

$$\begin{aligned}
&\frac{{\theta_k}^2}{2\alpha}\left[(z_k - x^*)^2 - (z_{k+1} - x^*)^2\right] \\
&\quad = \frac{{\theta_k}^2}{2\alpha}\left[(z_k - x^*)^2 - (z_k - \frac{\alpha}{\theta_k}\nabla f(y_k) - \frac{\alpha}{\theta_k}e_k - x^*)^2\right] \\
&\quad = -\frac{\alpha}{2}\left\|\nabla f(y_k)\right\|^2 - \frac{\alpha}{2}\left\|e_k\right\|^2 \\
&\qquad\qquad - \alpha\nabla f(y_k)e_k + \theta_k\nabla f(y_k)(z_k - x^*) + \theta_k e_k(z_k - x^*) \quad (23)
\end{aligned}$$

Hence, equation (22) can be rewritten as

$$\begin{aligned}
f(x_{k+1}) \leq\ & (1-\theta_k)f(x_k) + \theta_k f(x^*) + \frac{{\theta_k}^2}{2\alpha}\left[(z_k - x^*)^2 - (z_{k+1} - x^*)^2\right] \\
&+ \frac{\alpha}{2}\left\|\nabla f(y_k)\right\|^2 + \frac{\alpha}{2}\left\|e_k\right\|^2 + \alpha\nabla f(y_k)e_k - \theta_k e_k(z_k - x^*) \\
&- \alpha\left(1 - \frac{\alpha L}{2}\right)\left\|\nabla f(y_k)\right\|^2 - \alpha(1-\alpha L)\nabla f(y_k)e_k + \frac{\alpha^2 L}{2}\|e_k\|^2 \quad (24)
\end{aligned}$$

Taking expectations on both sides and using (11) and (12),

$$\begin{aligned}
E[f(x_{k+1})] \leq\ & (1-\theta_k)f(x_k) + \theta_k f(x^*) + \frac{{\theta_k}^2}{2\alpha}\left[(z_k - x^*)^2 - (z_{k+1} - x^*)^2\right] \\
&+ \frac{\alpha}{2}\left\|\nabla f(y_k)\right\|^2 + \frac{\alpha(C^2-1)}{2}\left\|\nabla f(y_k)\right\|^2 \\
&- \alpha\left(1 - \frac{\alpha L}{2}\right)\left\|\nabla f(y_k)\right\|^2 + \frac{\alpha^2 L(C^2-1)}{2}\|\nabla f(y_k)\|^2 \\
& \quad (25)
\end{aligned}$$

In the above equation, we see that for $\alpha = \frac{2-C^2}{LC^2}$, all the $\left\|\nabla f(y_k)\right\|^2$ terms cancel out. This value for α is valid for the condition that $0 < \alpha \leq \frac{2}{LC^2}$ as long as C^2 is between 0 and 2.

Now, (25) becomes

$$\begin{aligned}
E[f(x_{k+1})] \leq\ & (1-\theta_k)f(x_k) + \theta_k f(x^*) \\
&+ \frac{{\theta_k}^2 LC^2}{2(2-C^2)}\|z_k - x^*\|^2 - \frac{{\theta_k}^2 LC^2}{2(2-C^2)}\|(z_{k+1} - x^*)\|^2 \quad (26)
\end{aligned}$$

ACCEPTED MANUSCRIPT

Moving the last term on the right hand side to the left, introducing $f(x^*)$ on both sides and dividing throughout by θ_k^2,

$$\frac{1}{{\theta_k}^2}(E[f(x_{k+1})] - f(x^*)) + \frac{LC^2}{2(2-C^2)}\|z_{k+1} - x^*\|^2 \leq \frac{(1-\theta_k)}{{\theta_k}^2}(f(x_k) - f(x^*)) + \frac{LC^2}{2(2-C^2)}\|z_k - x^*\|^2 \quad (27)$$

Now, taking the expectation of (27) with respect to all the errors $\{e_0, e_1, ..., e_{k-1}\}$ recursively,

$$\frac{1}{{\theta_k}^2}(E[f(x_{k+1})] - f(x^*)) + \frac{LC^2}{2(2-C^2)}\|z_{k+1} - x^*\|^2 \leq \frac{(1-\theta_0)}{{\theta_0}^2}(E[f(x_0)] - f(x^*)) + \frac{LC^2}{2(2-C^2)}E\left[\|z_0 - x^*\|^2\right] \quad (28)$$

The second term on the left hand side is greater than zero, hence it can be omitted. Also, the value of $(1-\theta_0)/{\theta_0}^2$ is equal to zero, so the first term on the right hand side becomes zero. Replacing z_0 with x_0 and substituting $2/(k+2)$ for θ,

$$(E[f(x_{k+1})] - f(x^*)) \leq \frac{LC^2}{(2-C^2)(k+2)^2}E\left[\|x_0 - x^*\|^2\right] \quad (29)$$

This shows that this algorithm has a faster convergence rate of $O(1/k^2)$ compared to traditional online gradient method that has a convergence rate of $O(1/k)$. This is useful when models have to be learnt quickly in order to adapt to dynamic changes in the scene in the context of background modelling. Therefore, in the following section, we apply this type of updating scheme to RMoG.

ACCEPTED MANUSCRIPT

5. Regularisation for Region-based Mixture of Gaussians algorithm

In the RMoG algorithm [4], the EM method was used to derive the update equations. Since hard EM type classification is used to assign a pixel to a particular mixture, the non-convex cost function of MoG is converted into smaller convex optimisation problems [6] and hence, the momentum term can be applied to the update equations after assigning the pixel to a particular mixture. The online RMoG can be regularised by using an extrapolation involving the direction of the difference between the previous two values of the mixture parameter. Now, the question would be whether to apply it on all the parameters or apply it on certain parameters. The three parameters under consideration are the Means, Variances and Weights of the mixture components. The weights of the distribution are constrained by normalisation within a given neighbourhood, hence an additional regularisation term will have little or no influence on them. This can also be seen in the experiments from the simulated dataset of [6]. We also noticed empirically that applying the momentum term on the second order Variance term can cause the values to become negative at times thus rendering the system unstable. Even at times when the variance remained positive, it did not have a big influence on the performance. Therefore, we apply the regularisation only on the Mean update equation of the RMoG algorithm given in (2) which can be written as

$$\mu_{k,l,h}^{(t)} = (1-\rho)\mu_{k,l,h}^{(t-1)} + \rho\left(y_{i,j}^{(t)}\right) + \beta_t\left(\mu_{k,l,h}^{(t-1)} - \mu_{k,l,h}^{(t-2)}\right) \tag{30}$$

This application is justifiable because the mean parameter is the one that defines the cluster centers. In the case of background modelling, the

ACCEPTED MANUSCRIPT

means of the clusters are updated based on the pixel values and in regions having highly dynamic motion, the regularisation term helps smooth the mean parameter even if there is a high fluctuation of pixel colour samples.

The momentum parameter β_t is usually chosen to be between 0 and 1. The momentum term helps to find the optimum value faster, however, maintaining a large momentum value for a long period of time can cause the parameter estimates to diverge away from the optimum value. It is possible to control this divergence by increasing the momentum rate parameter from 0 to 1, as specified previously, for a time period proportional to the learning rate and then stopping, with subsequent updates taking place without the momentum term.

Algorithm Regularised Region-based Mixture of Gaussians

1. Consider a series of T images $Y = \{Y_1, Y_2, ..., Y_t, ..., Y_T\}$ where t is the index of the image at the current time instant. The image Y_t (of size $I \times J$ feature blocks) at location (i, j) can be denoted by $Y_t = \{\boldsymbol{y}_{i,j} : i = 1 : I, j = 1 : J\}$ where $\boldsymbol{y}_{i,j}$ are the different feature vectors. The model is given by the parameters $\{\theta_{t,i,j,h} : \mu_{t,i,j,h}, {\sigma^2}_{t,i,j,h}, \omega_{t,i,j,h}\}$ where $h = 1 : K$ is the index of the mixture and K is the total number of mixtures at each location. The size of each feature vector is dependent on the number of pixels in the feature blocks.

2. Initialise the first mixture of the model with the means equal to the pixel values of Y_1, variances are initialised by a suitable high value scaled by the number of pixels in each block and weights normalised over each region $\mathcal{R}_{i,j}$ given by r. Initialise the difference in the mean values to 0. Initialise

ACCEPTED MANUSCRIPT

the β value to 0.

3. For every subsequent image Y_t, calculate the most likely mixture $\theta_{t,k,l,h}$ where $(k,l) \in \mathcal{R}_{i,j}$ for the reference feature block $\boldsymbol{y}_{i,j}$ (over its entire neighbourhood). This can be calculated by using Euclidean distance.

4. Compare the distance of the most likely Gaussian mixture with its standard deviation scaled by a factor D. This indicates whether the pixel matches the mixture model or falls outside the model. The value of D was empirically chosen to be 2.5 in the experiments.

5. If a match is found, update the mixture parameters of the above Gaussian $\theta_{t,k,l,h}$ by using Eqn. (3) for the variance and Eqn. (30) for the mean with q corresponding to (k,l) and k corresponding to h.

6. Recalculate the differences in the mean values for the next image and update the beta value.

7. If no match is found, create a new Gaussian mixture if there aren't already K mixtures at the reference block location (i,j) or else replace the Gaussian mixture with the lowest weight (at the current block location (i,j)) with a new mixture by initialising it again. The new mixture is initialised by assigning the current pixel value as the mean, a small value comparable to the learning rate for the weight, and a large initial variance. This creates an arbitrary, wide Gaussian centered at the non-matching pixel value. The β value is reset for the new mixture t.

8. The weights are updated with Eqn. (4) and normalised so that they sum up to one.

ACCEPTED MANUSCRIPT

9. The background model is built by ranking the mixture components according to $\omega_{t,i,j,h}/\Sigma_{t,i,j,h}$ and selecting the top n ranked components such that their added ranking coefficients are greater than a certain threshold. If the observation falls within this model, it is classified as a background pixel; otherwise it is classified as foreground.

---

6. Experiments and Results

In this section, we first demonstrate the difference in performance of the online gradient method, with and without momentum, on data samples generated from a normal distribution. We then apply the algorithms to different types of mixture datasets to illustrate the influence of the momentum type updates. Then, we use momentum type updates in the regularised RMoG algorithm for subtracting dynamic backgrounds on well-known real world video sequences and compare its performance with that of the original MoG, regularised MoG and RMoG algorithms. Finally, we run the algorithm on the ChangeDetection.net benchmark [43] for comparison with other state of the art MoG-variant background subtraction algorithms.

6.1. Simulated Data Results

6.1.1. Univariate Data

First, we look at the convergence results of the online gradient method and the online gradient method with the additional momentum term for simulated data. These results can be seen in Figure 1. The data is generated from a Normal Distribution with a mean value of 128. We experimented

ACCEPTED MANUSCRIPT

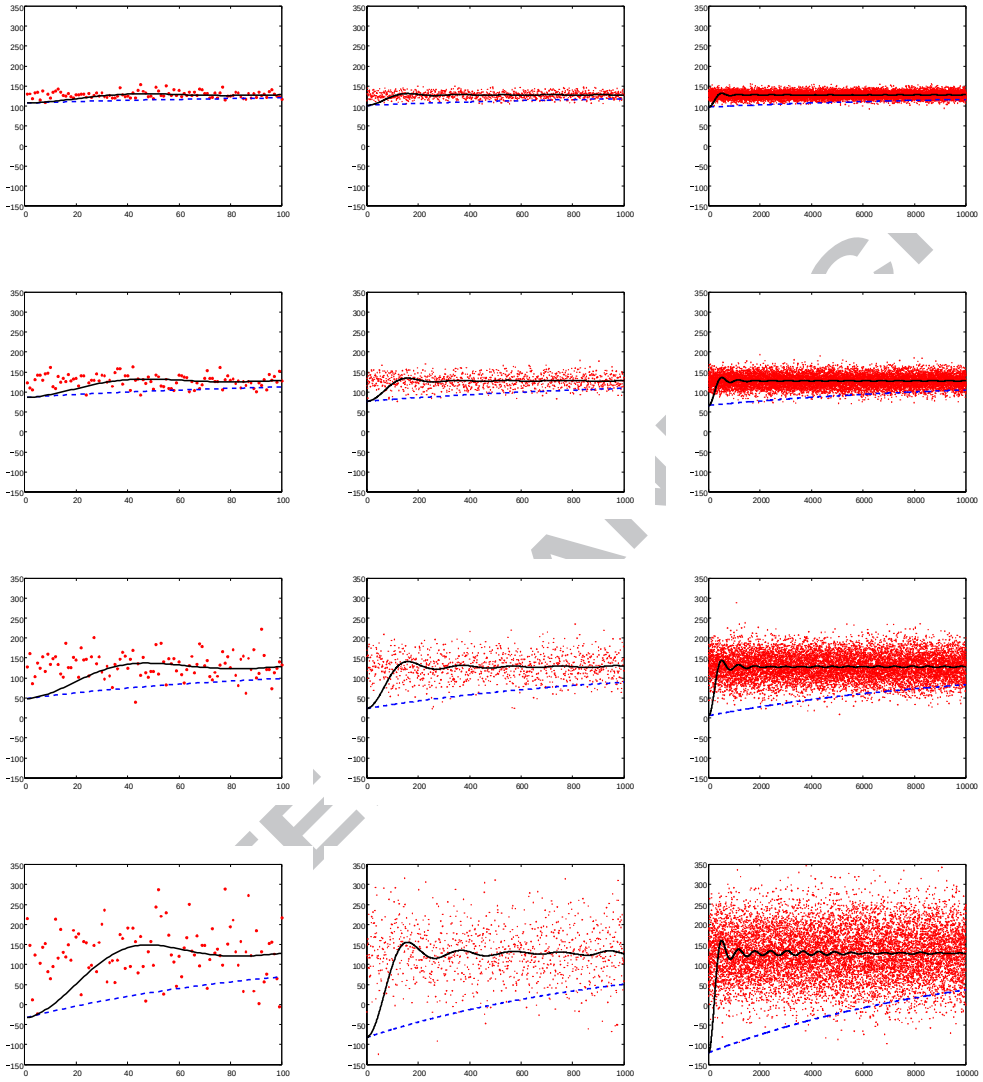


Figure 1: Convergence of Gradient methods without a momentum term (Blue/Dashed Line) and with a momentum term (Black/Solid line) for 100, 1000 and 10000 data samples from a Normal Distribution. First Row: Mean - 128, S.D. - 8; Second Row: Mean - 128, S.D. - 16; Third Row: Mean - 128, S.D. - 32; Fourth Row: Mean - 128, S.D. - 64.

ACCEPTED MANUSCRIPT

with different number of samples as well as different values of variances. The learning rate α in each case was fixed as $1/n$ where n is the number of samples. This value is small enough so that adding the momentum term does not cause the system to diverge. The learning rate is kept the same for both methods to highlight the contribution of the momentum term in online learning. In the context of online background modelling, the learning rate parameter determines the contribution of the new observation and is typically in the region of $1/T$ where T is the time period over which the scene changes. The momentum rate β was fixed as $(n-1)/(n+2)$ as mentioned in section 4. These experiments can be viewed as updating a single mixture component over time. The left column of Figure 1 shows the results for 100 data samples, the middle column shows the results for 1000 samples and the right column corresponds to the results of 10000 samples. The first row of Figure 1 corresponds to a standard deviation of 8, while subsequent rows correspond to increasing standard deviation values of 16, 32 and 64. In each case, the initial value was assigned to the lowest value from the data samples. Each set of experiments was run 100 times by randomly shuffling the data samples and the outputs are shown averaged over these 100 iterations.

In each of the plots, it can be seen that the algorithm with the momentum term (black/solid line) takes significant steps towards the optimum value 128 with the initial $50-100$ samples compared to the baseline approach without the momentum term (blue/dashed line). This improvement gets better with higher variances as the contribution from the momentum term is higher in these cases. This is particularly useful in the case of background modelling where regions with dynamic motion tend to have high variances in their pixel

ACCEPTED MANUSCRIPT

distributions. Adding the momentum term in the update equation helps learn these pixel variances faster compared to conventional type of updates without any added momentum.

6.1.2. Univariate Mixture Data

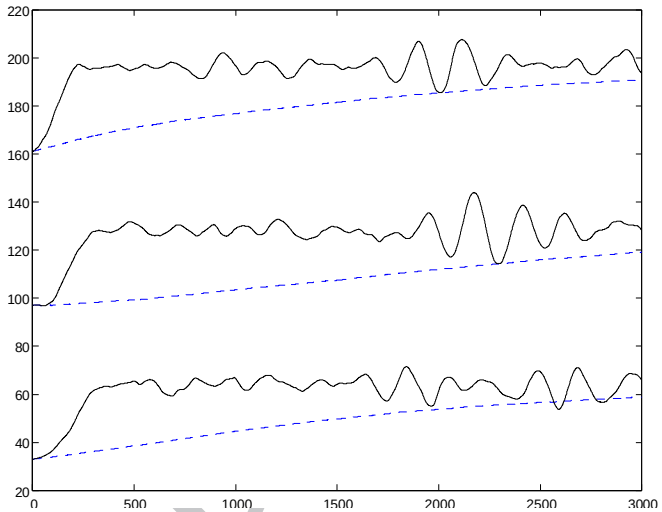


Figure 2: Convergence of means of 3 mixture data with equal weights $\mu = [64\ \ 128\ \ 192]$, $\sigma^2 = [256\ \ 256\ \ 256]$, $\omega = [0.33\ \ 0.33\ \ 0.33]$ for Standard Online MoG (Blue/Dashed Line) and MoG with Momentum (Black/Solid Line)

Next, we analysed the performance of online MoG with momentum type mean updates on simulated data. We generated a data of 3000 samples from a mixture distribution with means [64 128 192], variances [256 256 256] and mixture weights [1/3 1/3 1/3]. The classification of each sample to one of the mixtures is performed by matching based on Euclidean distance. The learning rate used in this experiment was empirically chosen as 0.001. Figure 2 shows the convergence of the mixture means over time. These results are

ACCEPTED MANUSCRIPT

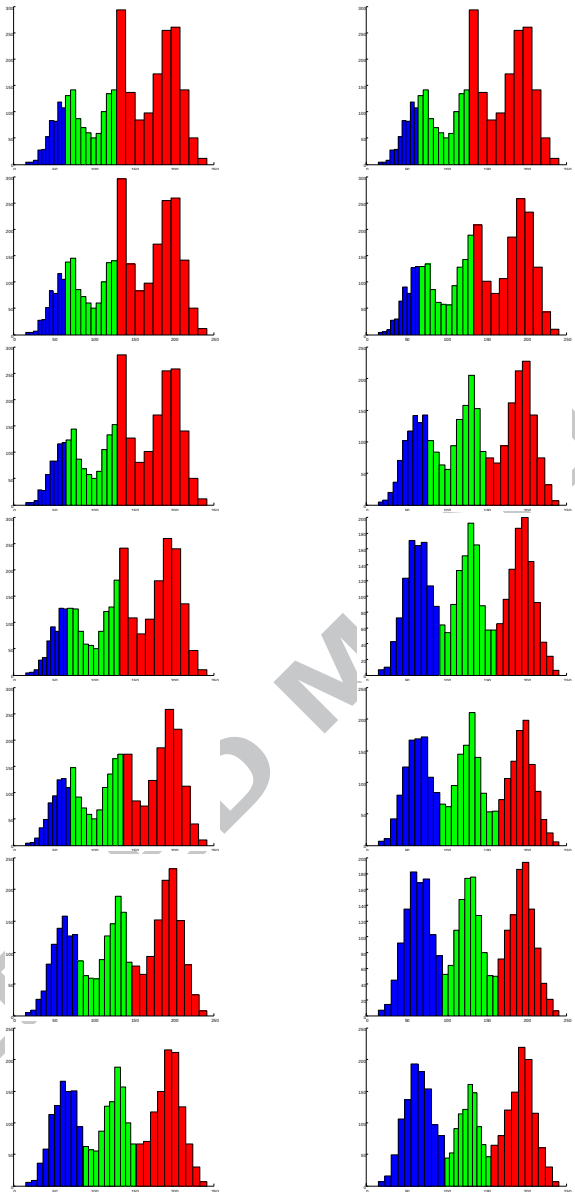


Figure 3: Clustering of test data with mixture distributions from Standard MoG (Left) and MoG with Momentum type updates (Right) updated after 1 (initial), 100, 200, 500, 1000, 2000 and 3000 (final) samples of the training set

ACCEPTED MANUSCRIPT

similar to the previous experiments in that the means of the momentum based algorithm converge much faster towards the optimum values compared to the algorithm without the use of momentum. For initialising the mixtures, the mean values were randomly chosen from the data samples, the initial covariances are a fraction of the mean global diagonal covariance matrix estimated from a fraction of the samples, and the weights were equally assigned to each of the mixtures in a manner similar to [44].

Now, in order to check the evolution of mixture components over time rather than just the mixture means, we clustered a test data of 3000 samples generated in a similar manner to the training data using the mixture distribution evolved at each time instant during the training phase. Figure 3 shows the histogram plots of the clustered data at different time instants. The first row shows the plot with the initial clusters and subsequent rows shows the plots clustered with the mixtures modelled after 100, 200, 500, 1000 and 2000 samples. The final row shows the histogram clustered with the mixtures updated with the entire training set. It is evident that these clusters are representative of the parameters used to generate the samples. We noticed that in the case of MoG with momentum type updates, the mixtures are able to cluster the data quite well after around 500 samples whereas the standard online MoG clusters the data accurately only after about 2000 samples.

6.1.3. Bivariate Mixture Data

We then analysed a three-component bivariate Gaussian mixture dataset [45] of 6000 samples with equal mixing weights, $\omega_1 = \omega_2 = \omega_3 = 1/3$, mean vectors, $\mu_1 = [-1 \;\; 1]^T$, $\mu_2 = [1 \;\; 1]^T$, $\mu_3 = [0 \;\; -1.414]^T$ and covariance matrices

ACCEPTED MANUSCRIPT

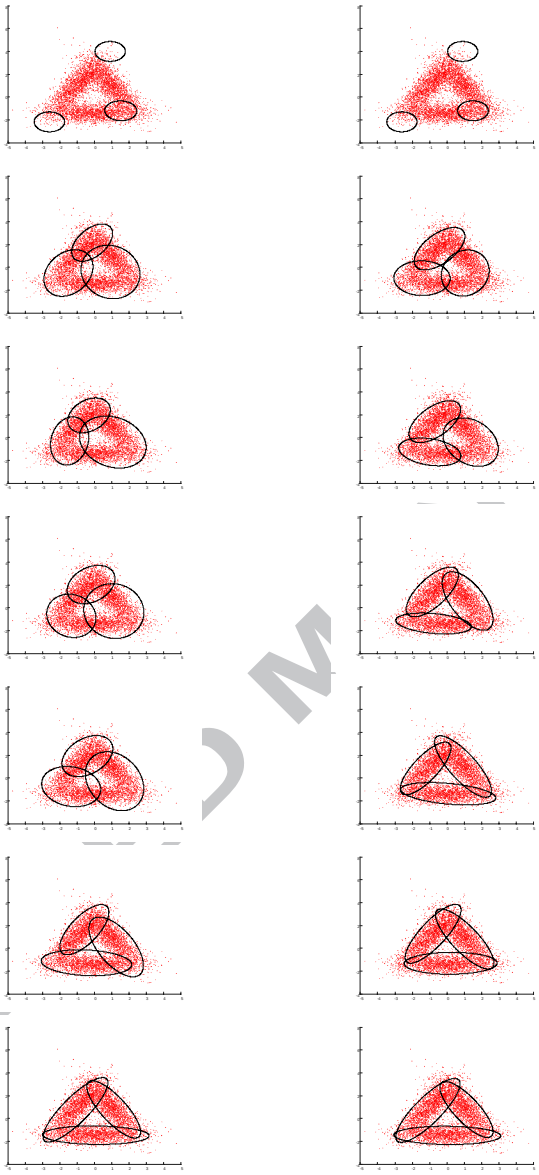


Figure 4: Clustering of the bivariate test data with the mixture distributions updated with Standard MoG (Left) and MoG with Momentum type updates (Right) after 1 (initial), 500, 1000, 1500, 2000, 3000 and 6000 (final) samples of the training set

ACCEPTED MANUSCRIPT

$\Sigma_1 = \begin{bmatrix} 0.65 & 0.7794 \\ 0.7794 & 1.55 \end{bmatrix}$, $\Sigma_2 = \begin{bmatrix} 0.65 & -0.7794 \\ -0.7794 & 1.55 \end{bmatrix}$, $\Sigma_3 = \begin{bmatrix} 2 & 0 \\ 0 & 0.2 \end{bmatrix}$. For this dataset, the learning rate was empirically chosen as 0.0067. The mixtures were initialised in the same way as the univariate mixture dataset.

Fig. 4 shows the clustering of test data, randomly generated using the same initial configuration, with the mixture components learnt after 1 (initial) sample, 500, 1000, 1500, 2000, 3000 and 6000 (final) samples of the training set. During the learning process, as the number of samples increases, the algorithms iteratively converge to the best fit for the mixture data. The momentum method accelerates this convergence, as can be seen from Fig. 4, and a good fit is obtained within 2000 samples of the test set. In contrast, it takes more than 3000 samples to converge to the best distribution for the standard approach.

6.1.4. Trivariate Mixture Data

Following univariate and bivariate datasets, we also evaluated the algorithm on a three-dimensional mixture dataset (Simulated Set 2 from [46], page 218) of 3000 samples. The mixing weights are $\omega_1 = 2/3$, $\omega_2 = 1/6$ and $\omega_3 = 1/6$, the mixture means are $\mu_1 = [0\ 0\ 0]^T$, $\mu_2 = [-6\ 3\ 6]^T$ and $\mu_3 = [6\ 6\ 4]^T$ and the full covariance matrices are given by $\Sigma_1 = \begin{bmatrix} 9 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $\Sigma_2 = \begin{bmatrix} 4 & -3.2 & -0.2 \\ -3.2 & 4 & 0 \\ -0.2 & 0 & 1 \end{bmatrix}$ and $\Sigma_3 = \begin{bmatrix} 4 & 3.2 & 2.8 \\ 3.2 & 4 & 2.4 \\ 2.8 & 2.4 & 2 \end{bmatrix}$.

Again, the initialisation of the Gaussian mixtures was done in the same manner as the two previous experiments. Figure 5 shows the clustering of test data of 3000 samples generated from the same mixture distribution with the mixtures learnt using the training data. A pattern similar to that encountered in both the univariate and bivariate datasets can be noticed here as well. Looking at the second row of Figure 5 (after 100 samples),

ACCEPTED MANUSCRIPT

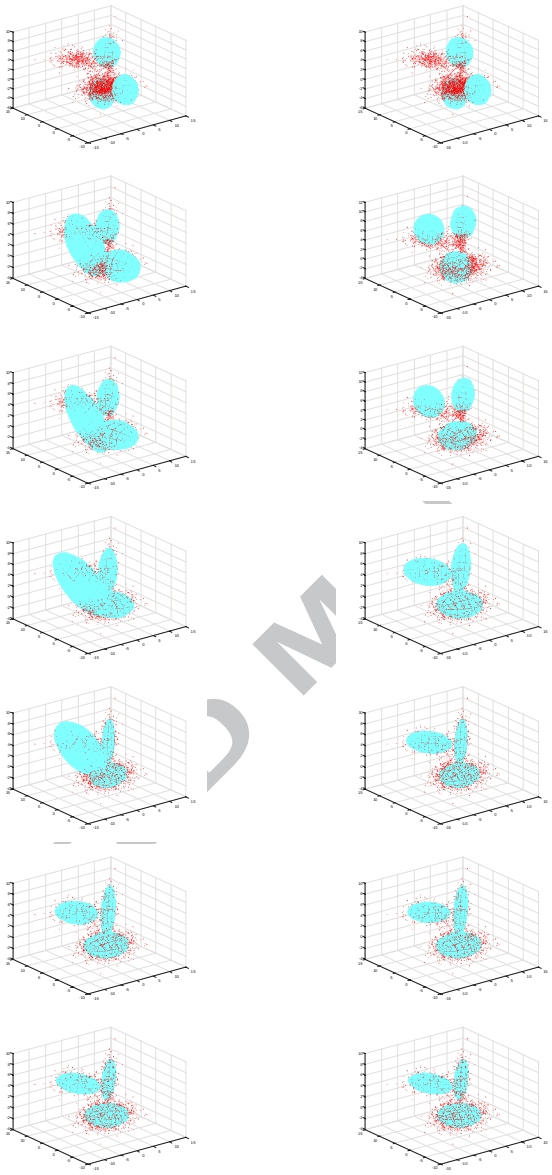


Figure 5: Clustering of the trivariate mixture data updated with Standard MoG (Left) and MoG with Momentum type updates (Right) after 1 (initial), 100, 200, 500, 1000, 2000 and 3000 (final) samples of the training set

ACCEPTED MANUSCRIPT

it is evident that the means are already closer to the actual values on the right side compared to the left. This shows that the momentum term pushes the initial means of the clusters towards their actual values faster than the normal MoG approach.

6.2. Evaluation of Regularised RMoG

In this section, we evaluated the regularised RMoG algorithm and compared it with MoG [2], regularised MoG [6] and RMoG [4]. For the evaluation we used four real world video sequences: the bottle sequence [47], the beach sequence [48], the waving trees sequence [49] and a CCTV sequence that was captured by us on board a moving bus. These sequences are widely used in the literature because of their dynamic backgrounds. The ROC curves for each of the sequences are shown in Figure 6. We show the results for two different values of the neighbourhood term r in the RMoG algorithm, r=1 and r=8. r=1 corresponds to the standard MoG approach while r=8 is the RMoG with 8×8 regions. We fix the neighbourhood size r as 8 because it is a standard neighbourhood size that can also be applied to background modelling in the compressed domain as in [50]. We also found empirically that r=8 produced the best results for the RMoG algorithm without compromising too much on the quality of the output. It is evident from the ROC curves that adding the momentum term increases the performance of the algorithm in both cases. In particular, the performance is better towards the upper left corner of the curve. This is the 'good' corner of the curve where the ratio of the true positive rate to the false positive rate is quite high. As the momentum based model learns faster than the baseline model, high variances in the dynamic background are learnt faster which we explain

ACCEPTED MANUSCRIPT

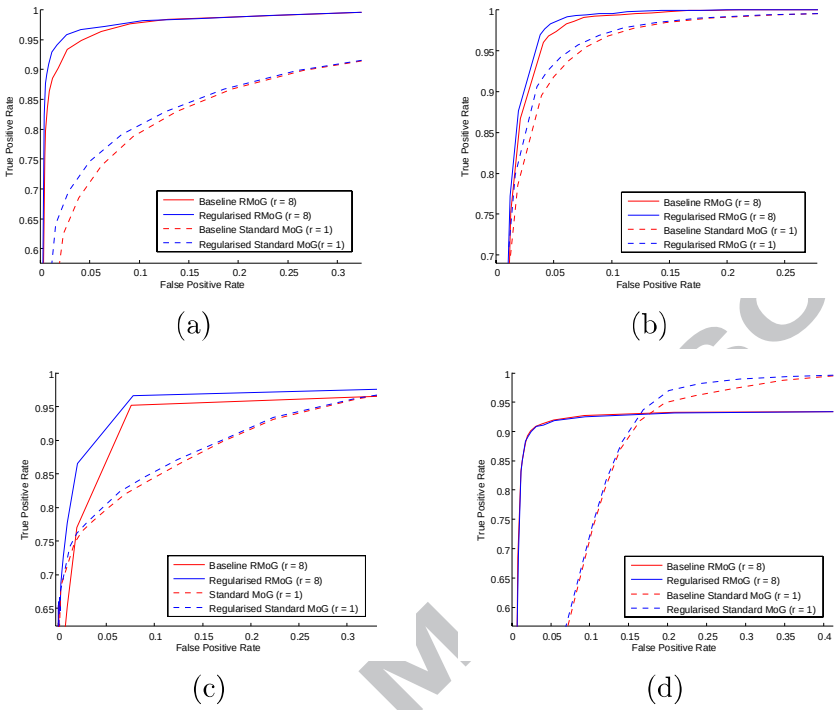


Figure 6: ROC curves, with, and without Regularisation, for the Bottle, (a), Bus, (b), Beach, (c), and Waving Trees, (d), sequences

in detail below with the help of Figure 7 by showing the distances between the means of the updated cluster and the pixel intensities at each instant. As was shown in the case of the simulated samples, with a learning rate of 0.001, the difference between the two approaches became more pronounced after about 100 samples.

An example output from each of the video sequences for different approaches are shown in Figure 8. In the bottle sequence, though the performance of RMoG is very good even without the momentum term, applying regularisation on the updates helps further reduce false positives in the im-

ACCEPTED MANUSCRIPT

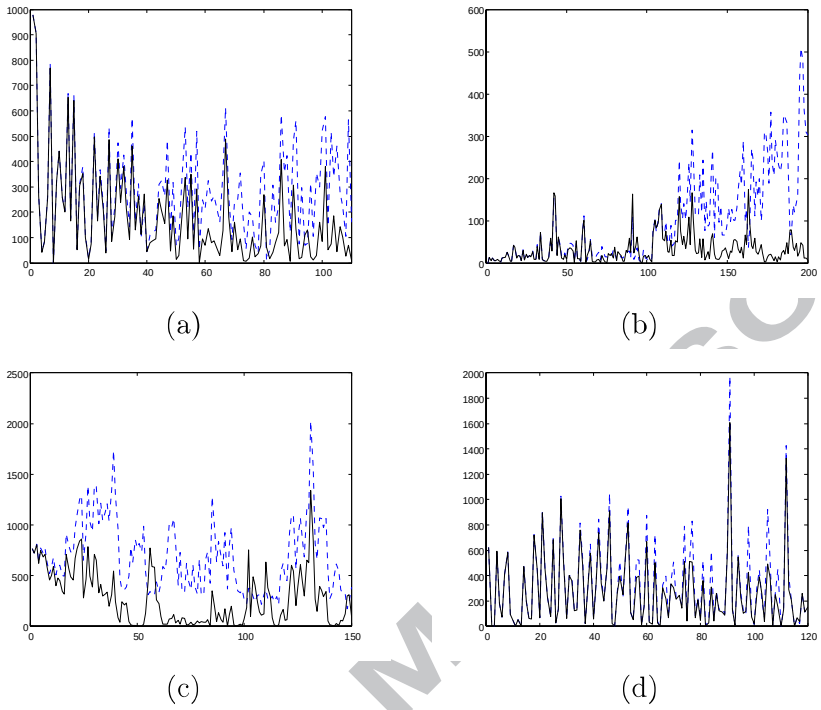


Figure 7: Squared Euclidean Distance between the pixel intensity and the mean of the closest cluster at every frame without the Momentum term (Blue/Dashed line) and with the Momentum term (Black/Solid Line) for the Bottle, (a), Bus, (b), Beach, (c), and Waving Trees, (d), sequences

ages. Not only does the regularisation term aid in reducing the false positives, but it also helps in reducing the false negatives in some cases as is evident from the output of the bus sequence and the beach sequence. Even in cases where the number of samples are small such as the waving trees sequence, adding the momentum term does not affect performance, with the regularised method working at least as well as the baseline method.

To take a closer look at why this momentum term helps improve the

ACCEPTED MANUSCRIPT

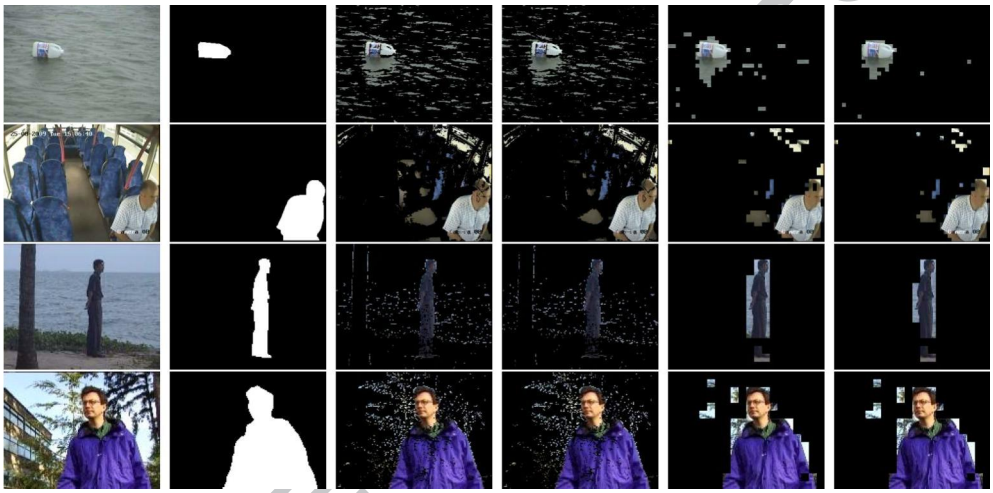


Figure 8: Sample outputs for different video sequences. First Column: Input frames; Second Column: Ground Truth; Third Column: Baseline MoG (r=1); Fourth Column: Regularised MoG (r=1); Fifth Column: Baseline RMoG (r=8); Sixth Column: Regularised RMoG (r=8)

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

performance of the algorithm, we studied the learning of the mixtures for individual pixels. One such pixel example from the dynamic background region of each video sequence is shown in Figure 7. Here, we plot the squared Euclidean distance used for classification between the pixel intensity value and the mean of the closest cluster for every frame. The black/solid lines represent the algorithm with the momentum term, while the blue/dashed lines represent the standard algorithm without the momentum term. These results for real data seem to follow a similar pattern as the results for the simulated data. While this distance is initially similar for both the methods, the difference between the distances from the two methods becomes more perceptible over time. This distance is generally lower for the algorithm with the additional momentum term while compared to the baseline algorithm thus indicating that the model learnt over time with the help of momentum gives a better representation of the underlying pixels. While these results aren't as good as the results from the simulated data, it has to be noted that the background subtraction algorithm involves a warm start of the mixtures which reduces the contribution from the difference term between successive iterates. In the waving trees sequence, there is a constant change between the mixtures chosen representing the sky and the leaves, therefore the number of times each mixture is updated is evenly split between them. This results in the momentum parameter not being updated enough times for the individual mixtures to produce a considerable difference between the two approaches. However, while learning each pixel, the momentum term helps to push the mixture mean closer to the pixel value compared to the update that does not involve the momentum term, thus reducing the distance between the mixture

ACCEPTED MANUSCRIPT

mean and the pixel intensity. The contribution from the momentum term is quite apparent after about 100 samples as can be seen in Fig. 7b and 7c. This follows the pattern noticed in the experiments with simulated data, where the difference between the two modelling techniques becomes quite noticeable after a few hundred samples.

6.3. Comparison with state of the art MoG variants

In this section we compared the regularised RMoG algorithm with MoG variants like the Stauffer and Grimson's MoG algorithm [2], KaewTraKulPong and Bowden's MoG algorithm [14] and Zivkovic's MoG algorithm [16]. For the evaluation, we used the widely used ChangeDetection.net(CDNet 2014) benchmark dataset [43]. This is arguably the most comprehensive publicly available benchmark consisting of a total of fifty-three video sequences that are partitioned into eleven categories, each of which poses a unique foreground detection challenge and contains four to six videos. The benchmark website maintains results in all categories for various state-of-the-art algorithms, along with a non-linear ranking algorithm that gets updated when the results of newer algorithms are added to the benchmark.

Table 1 shows the comparison of the F-measure scores for the regularised RMoG and some key background subtraction algorithms that use Gaussian mixture modelling. These include the seminal algorithm proposed by Stauffer and Grimson [2], the adaptive version of the algorithm introduced by Zivkovic [16], the MoG algorithm with two different up-date stages proposed by KaewTraKulPong and Bowden [14] and the RMoG [4], on which the current algorithm is based. In addition to the quantitative results for regularised RMoG, a typical output for one sequence from each of the categories

ACCEPTED MANUSCRIPT

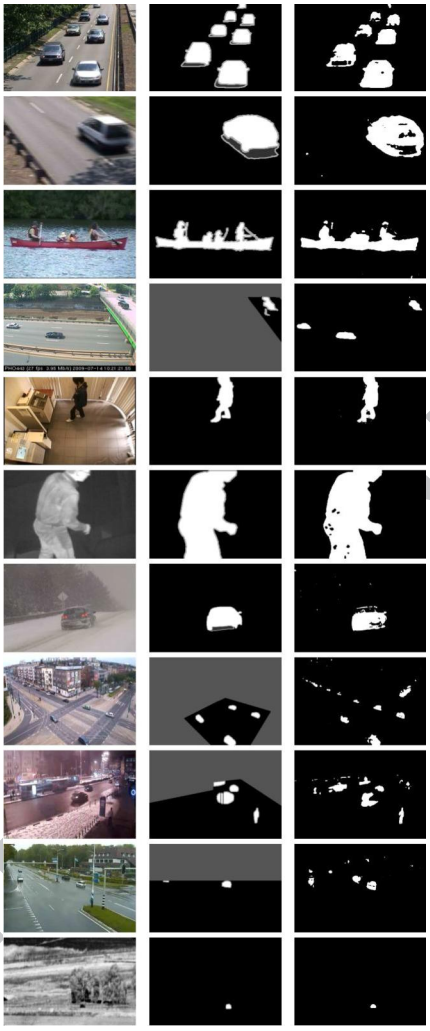


Figure 9: Typical Outputs of the Regularised RMoG algorithm for each category in CDNet 2014 (First Column: Input Frame, Second Column: Ground Truth). First Row: Baseline (highway); Second Row: Camera Jitter (traffic); Third Row: Dynamic Background (canoe); Fourth Row: Intermittent Object Motion (streetLight); Fifth Row: Shadow (copyMachine); Sixth Row: Thermal (diningRoom); Seventh Row: Bad Weather (snowFall); Eighth Row: Low Framerate (tramCrossroad_1fps); Ninth Row: Night Videos (tramStation); Tenth Row: PTZ (twoPositionPTZCam); Eleventh Row: Turbulence (turbulence2)

ACCEPTED MANUSCRIPT

Category	MoG (S&G)	MoG (ZZ)	MoG (K&B)	Regularised RMoG
BL	0.82	**0.84**	0.71	0.80
CJ	0.60	0.57	0.58	**0.72**
DB	0.63	0.63	0.67	**0.73**
IOM	0.52	0.53	0.39	**0.57**
S	**0.74**	0.73	0.71	0.72
Th	**0.66**	0.65	0.48	0.48
BW	**0.74**	**0.74**	0.61	0.71
LF	**0.54**	0.51	0.37	0.53
NV	**0.41**	0.40	0.34	**0.41**
PTZ	0.15	0.10	0.15	**0.26**
Tu	0.47	0.42	**0.57**	0.56
Overall	0.57	0.56	0.51	**0.59**

Table 1: Comparison of F-Measures for all the categories in CDNet 2014 Benchmark for the different types of MoG based background subtraction algorithms. MoG (S&G) - Stauffer and Grimson's MoG algorithm [2], MoG (ZZ) - Zoran Zivkovic's MoG algorithm [16], MoG (K&B) - KaewTraKulPong and Bowden's MoG algorithm [14], Regularised RMoG - This paper.

is shown in Fig. 9. A 5×5 median filter has been applied to these images as a post-processing step.

For the Baseline (BL) category, the algorithm performs reasonably well with an F-measure of 0.80. From Fig. 9 (1st row, 3rd column), the segmented foreground objects are very well defined with very few background

ACCEPTED MANUSCRIPT

false positives. This is further reflected in an average precision of 0.91 for the BL sequences, and a low false positive rate of 0.002. The next two categories, Camera Jitter (CJ) and Dynamic Background (DB), are the most relevant in relation to the regularised RMoG algorithm. In the CJ sequence, the scene background blurs, due to an unstable camera, Fig. 9 (2nd Row, 1st column). Our algorithm handles this quite well, as can be seen in the segmented output image on the same row, third column in Fig. 9. The F-Measure for the CJ sequences is 0.72, which is the best of all the evaluated algorithms in Table 1. The next category is that of DBs, which includes DBs such as waving trees and rippling water, Fig. 9 (third row). The F-Measure for the DB sequences is 0.73, which is significantly better than for the other algorithms whose scores are 0.63 and 0.67. The performance in these two categories show that the regularised RMoG is the best in handling DBs.

The Intermittent Object Motion (IOM) category is one of the most difficult in the benchmark, consisting of scenes where moving objects become static for a while and then start moving again. While the IOM F-Measure for the regularised RMoG is not very high, 0.57, it is still better than all the other MoG-variant algorithms. The Shadow category involves scenes with dark shadows falling on the background, Fig. 9 (5th row, 1st column). In this case, the original MoG approach performs marginally better than the regularised RMoG, however, the average precision for the latter, 0.80, is much higher than that obtained for the former, 0.72. This is reflected in the segmented output, Fig. 9 (5th row, 3rd column), which shows good foreground definition and no false alarms. In contrast to the previous categories, we noticed that the regularised RMoG algorithm did not perform well in the

ACCEPTED MANUSCRIPT

Thermal category, with a relatively low F-Measure of 0.48. This is because the intensity images generated by the far infrared cameras, Fig. 9 (6th row, 1st column), result in a very low recall of 0.35. Further evidence for this is the holes in the foreground region, Fig. 9 (6th row, 3rd column).

The next category is of Bad Weather (BW) conditions, which includes scenes taken during cold winter conditions with blizzards or heavy snowfall, Fig. 9 (7th row, 1st column). The F-Measure of our algorithm is slightly lower than that for both MoG (ZZ) and MoG (S&G), however, its precision, 0.91, is much higher. Again, this is reflected in the good foreground definition and lack of background false positives in Fig. 9 (7th row, 3rd column). For scenes captured with low frame cameras, Fig. 9 (8th row, 1st column), the F-measures for regularised RMoG, 0.53, is slightly lower than MoG(S&G) on 0.54. The Night Video is another quite challenging category due to the presence of car headlights, Fig. 9 (9th row, 1st column). In this case, the F-measures for regularised RMoG and MoG (S&G) are joint highest on 0.41.

The Pan-Tilt-Zoom (PTZ) category, Fig. 9 (10th row, 1st column), is probably the most challenging in the benchmark, especially since most of the background subtraction algorithms have been designed for a fixed camera. Our algorithm outperforms the others, with an F-Measure of 0.26. The last category contains videos that show air turbulence (Tu) due to heat, Fig. 9 (11th row, 1st column). In this case, our algorithm has an F-Measure of 0.56, which is only marginally smaller than the best, MoG (K&B) with 0.57. Fig. 9 (11th row, 3rd column) shows that our algorithm is able to subtract the turbulent background very effectively with no false positives.

In summary, the average F-Measure over all eleven categories for our

ACCEPTED MANUSCRIPT

algorithm is 0.59, which is the best compared to the other MoG-variants. The regularised RMoG performs best in four of the categories, is joint top in another and second in two others. The only category for which performance is relatively poor is the Thermal category. This is further reflected in the segmented output images in Fig. 9 (3rd column), which demonstrate, in the main, good foreground definition with very low background false positives. The results are particularly good for those categories that do not have static backgrounds; CJ, DB, IOM, PTZ and Tu. These results show that the regularised RMoG algorithm produces better results than the other state of the art MoG-variants evaluated here.

7. Conclusion

To summarise, we derived the proof for the $O(1/k^2)$ convergence rate of the online version of the gradient method with heavy ball-like updates. Following this, we applied this method to the RMoG algorithm and evaluated against an extensive real-world video dataset containing numerous dynamic backgrounds. The results demonstrate that the algorithm outperforms other state of the art MoG variants. In conclusion, the momentum based updates increase the convergence rate, thereby improving the performance of the RMoG algorithm.

We believe the Regularised RMoG algorithm shows good potential for robust deployment in real-world practical applications. Future work will involve investigating the critical limits for the learning rate and also investigating third and higher order steps in the learning method.

ACCEPTED MANUSCRIPT

Appendix: Proof of Convergence of the Online Gradient Method with Momentum Term

Here, we show the proof of convergence of the online gradient method with a momentum term [8]. The algorithm is usually of the form

$$y_k = x_k + \beta_k(x_k - x_{k-1}) \tag{31}$$

$$x_{k+1} = y_k - \alpha \nabla_{\omega_k} f(y_k) \tag{32}$$

Notice the subscript ω for the gradient that indicates that the gradient is equivalent to the expectation of the current observation error i.e. $\nabla_\omega f(y_k) = E(H(\omega_k, y_k))$. Here, ω_k are the observations at each instant k.

In order to prove the convergence, the Supermartingale Convergence Theorem [40] is first introduced.

Lemma 1 (Supermartingale Convergence Theorem) Let Y_k, Z_k and W_k, $k = 0, 1, 2, ...,$ be three sequences of random variables and let F_k, $k = 0, 1, 2, ...,$ be sets of random variables such that $F_k \subset F_{k+1}$ for all k. Suppose that:

1. The random variables Y_k, Z_k and W_k are non-negative, and are functions of the random variables in F_k.
2. For each k, we have

$$E\{Y_{k+1}|F_k\} \leq Y_k - Z_k + W_k$$

3. There holds, with probability 1, $\sum_{k=0}^{\infty} W_k < \infty$

Then, we have $\sum_{k=0}^{\infty} Z_k < \infty$, and the sequence Y_k converges to a nonnegative random variable Y, with probability 1. ■

ACCEPTED MANUSCRIPT

For the online gradient method with the momentum term to converge, we assume the following:

1. f is convex, differentiable and finite for all x
2. There exists a finite solution x^*
3. The gradient is Lipschitz continuous with a constant L, i.e.

$$||\nabla f(x) - \nabla f(y)||_2 \leq L||x - y||_2 \qquad \forall x, y \tag{33}$$

These conditions provide the bounds necessary for f. The first assumption gives a linear lower bound on f

$$f(y) \geq f(x) + \nabla f(x)^T(y - x) \qquad \forall x, y \tag{34}$$

The third assumption provides a quadratic upper bound on f

$$f(y) \leq f(x) + \nabla f(x)^T(y - x) + \frac{L}{2}||x - y||_2^2 \tag{35}$$

The proof continues by expanding the term for x_{k+1} and setting the upper limit

$$\begin{aligned} f(x_{k+1}) &= f(y - \alpha \nabla_{\omega_k} f(y)) \\ &= f(y - \alpha E(H(\omega_k, y_k))) \\ &\leq f(y) + \nabla f_{\omega_k}(y)^T(y - \alpha \nabla f_{\omega_k}(y) - y) \\ &\quad + \frac{L}{2}||y - \alpha E(H(\omega_k, y_k)) - y||_2^2 \\ &= f(y) - \alpha||\nabla f_{\omega_k}(y)||_2^2 + \frac{L\alpha}{2}E(H(\omega_k, y_k)^2) \end{aligned} \tag{36}$$

The first inequality is by using the quadratic upper bound due to the third assumption. Applying conditional expectation on both sides,

$$E(f(x_{k+1})) \leq E(f(y)) - \alpha||\nabla f_{\omega_k}(y)||_2^2 + \frac{L\alpha}{2}E(H(\omega_k, y_k)^2) \tag{37}$$

ACCEPTED MANUSCRIPT

Now, Lemma 1 can be applied to this equation as we are dealing with positive values as long as the final term of the equation converges with a probability of 1. Assuming $\alpha = \frac{1}{L}$, it is enough to show that the term $E(H(\omega_k, y_k)^2)$ is bounded.

In online mode, this term refers to the second moment of the updates [5] and can be decomposed as

$$E(H(\omega_k, y_k)^2) = (\nabla_{\omega_k} f(y))^2 + var_{\omega_k} H(\omega_k, y_k) \tag{38}$$

This second term, given by the variance, indicates the noise due to the stochastic nature of the algorithm. The term remains positive at all locations, even at the optimum value. Since the gradient is Lipschitz continuous, it imposes the bound required for the almost sure convergence of the final term of Equation (37).

Therefore, by using Lemma 1, it can be said that

$$x_k \xrightarrow[k\to\infty]{a.s.} x^* \tag{39}$$

This proves that the online gradient method with momentum almost surely converges to an optimum value with the probability of 1.

References

[1] A. Sobral, A. Vacavant, A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos, Computer Vision and Image Understanding 122 (2014) 4–21.

[2] C. Stauffer, W. E. L. Grimson, Adaptive background mixture models for real-time tracking, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2246–2252.

ACCEPTED MANUSCRIPT

[3] H. Robbins, S. Monro, A stochastic approximation method, The Annals of Mathematical Statistics 22 (1951) 400–407.

[4] S. Varadarajan, P. Miller, H. Zhou, Spatial mixture of gaussians for dynamic background modelling, in: Advanced Video and Signal Based Surveillance (AVSS), 2013 10th IEEE International Conference on, pp. 63–68.

[5] L. Bottou, Online algorithms and stochastic approximations, in: Online Learning and Neural Networks, Cambridge University Press, 1998.

[6] H. Wang, P. Miller, Regularized online mixture of gaussians for background subtraction, in: 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 249–254.

[7] Y. Nesterov, Introductory lectures on convex optimization. a basic course, 2004.

[8] S. Varadarajan, H. Wang, P. Miller, H. Zhou, Regularised region-based mixture of gaussians for dynamic background modelling, in: Advanced Video and Signal Based Surveillance (AVSS), 2014 11th IEEE International Conference on, pp. 19–24.

[9] T. Bouwmans, Recent advanced statistical background modeling for foreground detection: A systematic survey, Recent Patents on Computer Science 4 (2011).

[10] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland, Pfinder: Real-time tracking of the human body, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (1997) 780–785.

ACCEPTED MANUSCRIPT

[11] N. Friedman, S. Russell, Image segmentation in video sequences: a probabilistic approach, in: Proceedings of the Thirteenth conference on Uncertainty in artificial intelligence, pp. 175–181.

[12] R. M. Neal, G. E. Hinton, A New View of the EM Algorithm that Justi es Incremental and Other Variants, in: Learning in Graphical Models, pp. 355–368.

[13] S. J. McKenna, Y. Raja, S. Gong, Tracking colour objects using adaptive mixture models, Image and Vision Computing 17 (1999) 225–231.

[14] P. KaewTraKulPong, R. Bowden, An improved adaptive background mixture model for real-time tracking with shadow detection, in: Advanced Video-Based Surveillance Systems, Springer, 2001, pp. 135–144.

[15] D.-S. Lee, Effective gaussian mixture learning for video background subtraction, Pattern Analysis and Machine Intelligence, IEEE Transactions on 27 (2005) 827–832.

[16] Z. Zivkovic, Improved adaptive gaussian mixture model for background subtraction, in: Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on, volume 2, pp. 28–31.

[17] A. Elgammal, R. Duraiswami, D. Harwood, L. S. Davis, Background and foreground modeling using nonparametric kernel density estimation for visual surveillance, in: PROCEEDINGS OF THE IEEE, pp. 1151–1163.

[18] H. Greenspan, J. Goldberger, A. Mayer, Probabilistic space-time video

ACCEPTED MANUSCRIPT

modelling via piecewise gmm, Pattern Recognition and Machine Intelligence, IEEE Transactions on 26 (2004) 384–396.

[19] Y. Sheikh, M. Shah, Bayesian modeling of dynamic scenes for object detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2005) 1778–1792.

[20] P.-M. Jodoin, M. Mignotte, J. Konrad, Statistical background subtraction using spatial cues, IEEE Transactions on Circuits and Systems for Video Technology 17 (2007) 1758–1763.

[21] S. Zhang, H. Yao, S. Liu, Spatial-temporal nonparametric background subtraction in dynamic scenes, in: Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on, pp. 518–521.

[22] G. Dalley, J. Migdal, W. E. L. Grimson, Background subtraction for temporally irregular dynamic textures, in: IEEE Workshop on Applications of Computer Vision, pp. 1–7.

[23] T. Yu, C. Zhang, M. Cohen, Y. Ruy, Y. Wu, Monocular video foreground/background segmentation by tracking spatial-color gaussian mixture models, in: Proceedings of the IEEE Workshop on Motion and Video Computing.

[24] B. T. Polyak, Introduction to Optimization, Optimization Software, Inc., Publications Divisions, New York, 1987.

[25] A. Beck, M. Teboulle, A fast iterative shrinkage thresholding algorithm for linear inverse problems, SIAM Journal on Imaging Sciences 2 (2009) 183–202.

ACCEPTED MANUSCRIPT

[26] P. Tseng, On accelerated proximal gradient methods for convex-concave optimization, SIAM Journal on Optimization (2008).

[27] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1, MIT Press, 1986, pp. 318–362.

[28] J. Shynk, S. Roy, The lms algorithm with momentum updating, in: Circuits and Systems, 1988., IEEE International Symposium on, pp. 2651–2654.

[29] D. P. Bertsekas, A new class of incremental gradient methods for least squares problems, SIAM Journal on Optimization 7 (1996) 913–926.

[30] N. Zhang, An online gradient method with momentum for two-layer feedforward neural networks, Applied Mathematics and Computation 212 (2009) 488–498.

[31] C. Bao, Y. Wu, H. Ling, H. Ji, Real time robust l1 tracker using accelerated proximal gradient approach, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pp. 1830–1837.

[32] S. V. N. Vishwanathan, N. N. Schraudolph, M. W. Schmidt, K. P. Murphy, Accelerated training of conditional random fields with stochastic gradient methods, in: Proceedings of the 23rd International Conference on Machine Learning, pp. 969–976.

[33] H. Le, L. Hu, Y. Feng, Momentum based level set method for accurate object tracking, International Journal of Intelligent Systems and Applications 2 (2010) 10–16.

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

[34] T. Andersson, G. Lathen, R. Lenz, M. Borga, Modified gradient search for level set based image segmentation, IEEE Transactions on Image Processing 22 (2013) 621–630.

[35] H. Wang, P. Miller, Scaled heavy-ball acceleration of the richardson-lucy algorithm for 3d microscopy image restoration, IEEE Transactions on Image Processing 23 (2014) 848–854.

[36] J. Domke, Generic methods for optimization-based modeling, in: Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics (AISTATS-12), pp. 318–326.

[37] D. Li, L. Xu, E. Goodman, A fast foreground object detection algorithm using kernel density estimation, in: Signal Processing (ICSP), 2012 IEEE 11th International Conference on, volume 1, pp. 703–707.

[38] C. Guyon, T. Bouwmans, E.-H. Zahzah, Foreground detection by robust pca solved via a linearized alternating direction method, in: Image Analysis and Recognition, Springer, 2012, pp. 115–122.

[39] N. Greggio, A. Bernardino, C. Laschi, P. Dario, J. Santos-Victor, Self-adaptive gaussian mixture models for real-time video segmentation and background subtraction, in: IEEE 10th International Conference on Intelligent System Design and Applications.

[40] D. P. Bertsekas, Convex optimization theory by supplementary chapter 6 on convex optimization algorithms, 2013.

[41] M. Schmidt, N. Le Roux, Fast convergence of stochastic gradient descent under a strong growth condition.

ACCEPTED MANUSCRIPT

[42] P. Tseng, An incremental gradient(-projection) method with momentum term and adaptive stepsize rule, SIAM Journal on Optimization 8 (1998) 506–531.

[43] Y. Wang, P.-m. J. Fatih, P. Janusz, K. Yannick, B. Prakash, Change Detection 2014 Benchmark, 2014.

[44] M. A. Figueiredo, A. Jain, Unsupervised learning of finite mixture models, Pattern Analysis and Machine Intelligence, IEEE Transactions on 24 (2002) 381–396.

[45] T. Huang, H. Peng, K. Zhang, Model selection for gaussian mixture models.

[46] G. McLachlan, D. Peel, Finite Mixture Models, Wiley Series in Probability and Statistics, Wiley-Interscience, 2000.

[47] J. Zhong, S. Sclaroff, Segmenting foreground objects from a dynamic textured background via a robust kalman filter, in: Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on, pp. 44–50.

[48] L. Li, W. Huang, I. Y. H. Gu, Q. Tian, Foreground object detection from videos containing complex background, in: Proceedings of the eleventh ACM international conference on Multimedia, pp. 2–10.

[49] K. Toyama, J. Krumm, B. Brumitt, B. Meyers, Wallflower: Principles and practice of background maintenance, in: Seventh International Conference on Computer Vision, pp. 255–261.

ACCEPTED MANUSCRIPT

[50] S. Popa, D. Crookes, P. Miller, Hardware acceleration of background modeling in the compressed domain, Information Forensics and Security, IEEE Transactions on 8 (2013) 1562–1574.

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

Fast Convergence of Regularised Region-based Mixture of Gaussians for Dynamic Background Modelling

Sriram Varadarajan, Hongbin Wang, Paul Miller and Huiyu Zhou

HIGHLIGHTS

- Derivation of $O(1/k^2)$ rate of convergence for Online Gradient Method with Momentum
- Novel foreground detection algorithm for dynamic background modelling
- Fast Convergence of Mixtures demonstrated on various sets of simulated data
- Superior performance compared to other state-of-the-art algorithms on real videos

ACCEPTED MANUSCRIPT